



**Shanghai Jiao Tong University**  
**Software Theory and Practice Group**

# Understanding Myths and Realities of Test-Suite Evolution

FSE 2012, Leandro Sales Pinto, Saurabh Sinha, Alessandro Orso

Jinbo Du, 2013/10/11  
marstone@sjtu.edu.cn

# Abstract

- Test suites evolve throughout their lifetime
- provide initial insight on how test cases are **added**, **removed**, and **modified** in practice
- Without such knowledge, we risk to develop techniques that may work well for only a small number of tests or, worse, that may not work at all in most realistic cases

# Definitions and Terminology

- A system  **$S = (P, T)$**  consists of a program P and a test suite T
- A test suite  **$T = \{t_1, t_2, \dots, t_n\}$**  consists of a set of unit test cases.
- **Test(P, t)** is a function that executes test case t on program P and returns the outcome of the test execution
- The outcome of the test execution:
  - **Pass**: The execution of P against t succeeds.
  - **FailCE**: fails because a class or method accessed in t does not exist in P.
  - **FailRE**: fails due to an exception
  - **FailAE**: fails due to an assertion violation.

# Scenarios 1 - Test modified

- Given a system  $S = (P, T)$ , a modified version of  $S$ ,  $S' = (P', T')$
- Scenarios 1: Test  $t$  exists in  $S$  and  $S'$  and is modified.
- **Category [TESTREP] : Repaired Test**
  - $\text{Test}(P', t) = \text{Fail} \wedge \text{Test}(P', t') = \text{Pass}$
  - $t$  is repaired
- **Category [TESTMODNOTREP]: Refactored(Modified) Test**
  - $\text{Test}(P', t) = \text{Pass} \wedge \text{Test}(P', t') = \text{Pass}$
  - $t$  is refactored, updated to test a different scenario, or is made more/less discriminating



# Scenarios 2 - Test removed

- Scenarios 2: Test  $t$  is removed in  $S'$
- [TESTDEL(AE | RE)]: Hard-To-Fix Tests
  - $\text{Test}(P', t) = \text{FailRE} \mid \text{FailAE}$
  - $t$  is too difficult to fix
- [TESTDEL(CE)]: Obsolete Tests
  - $\text{Test}(P', t) = \text{FailCE}$
  - $t$  is obsolete or is too difficult to fix
- [TESTDEL(P)]: Redundant Tests
  - $\text{Test}(P', t) = \text{Pass}$
  - $t$  is redundant

# Scenarios 3 - Test added

- Scenarios 3 :Test  $t'$  is added in  $S'$
- [TESTADD(AE|RE)] : Bug-Fix Tests
  - $\text{Test}(P, t') = \text{FailRE} \mid \text{FailAE}$
  - $t'$  is added to validate a bug fix
- [TESTADD(CE)] : New Feature Tests
  - $\text{Test}(P, t') = \text{FailCE}$
  - $t'$  is added to test a new functionality or a code refactoring
- [TESTADD(P)] : Coverage-Augmentation Tests
  - $\text{Test}(P, t') = \text{Pass}$
  - $t'$  is added to test an existing feature or for coverage-based augmentation

# Empirical Study

Program	First	Last	Versions	Testcases
Commons Lang	2002	2011	11	2531
Commons Math	2004	2012	7	3164
Gson	2008	2011	16	1200
JFreeChart	2006	2011	12	2211
JodaTime	2004	2012	27	3963
PDM	2004/01	2004/12	17	573
Total			88	17427

# Test repair

Q1. How often do the different categories of test-suite changes that we consider occur?

TESTREP	TESTMOD NOTREP	TESTDEL (AE RE)	TESTDEL (CE)	TESTDEL (P)	TESTADD (AE RE)	TESTADD (CE)	TESTADD (P)
1121	3990	112	1482	947	1358	6753	1664
(6.4%)	(22.9%)	(0.6%)	(8.5%)	(5.4%)	(7.8%)	(38.8%)	(9.5%)

- Test repairs are only part of the story: test repair, and test modifications in general, are a minority of all test changes.
- Test repairs occur often enough in practice to justify the development of automated repair techniques.



# Test Modification

Q2. How often do test repairs involve modifications to existing assertions only? How often they require more complex modifications of the tests instead?

Call Add	Call Del	Param Add	Param Del	Param Mod	Assert Add	Assert Del	Assert Mod	Value Mod
530	530	40	540	118	233	271	37	61
47.3%	47.3%	3.6%	48.2%	10.5%	20.8%	24.2%	3.3%	5.4%

- Test-repair techniques that focus exclusively on assertions can target only a small subset of all broken (repairable) test cases and must be extended to achieve a wider applicability of automated test repair
- Investigating techniques for automated support of non-repair changes does not appear to be a promising research avenue. These techniques would require considerable manual guidance and may end up being similar to traditional refactoring tools.

# Test Deletion

Q3. Why are tests deleted? Are tests in different categories deleted for different reasons?

Program	$\text{TESTDEL}_{(P)}$	Same Branch Coverage	Reduced Branch Coverage
Commons Lang	122	120 (98.4%)	2 (1.6%)
Commons Math	128	50 (39.1%)	78 (60.9%)
Gson	143	140 (97.9%)	3 (2.1%)
JFreeChart	19	15 (78.9%)	4 (21.1%)
JodaTime	58	46 (79.3%)	12 (20.7%)
PMD	477	341 (71.5%)	136 (28.5%)
Total	947	712 (75.2%)	235 (24.8%)

- Tests that fail in the new version of a program—because of a compilation error, a runtime exception, or a failed assertion—tend to be deleted not because they are difficult to repair, but because they are obsolete.
- Tests that would pass in the new version of a program but appear to have been deleted have, in most cases, simply been moved or renamed.

# Test Addition

Q4. Why are tests added? Are tests in different categories added for different reasons?

Program	TESTADD <sub>(P)</sub>	Same Branch Coverage	Increased Branch Coverage
Commons Lang	334	139 (41.6%)	195 (58.4%)
Commons Math	147	99 (67.3%)	48 (32.7%)
Gson	363	155 (42.7%)	208 (57.3%)
JFreeChart	277	162 (58.5%)	115 (41.5%)
JodaTime	271	198 (73.1%)	73 (26.9%)
PMD	272	177 (65.1%)	95 (34.9%)
Total	1664	930 (55.9%)	734 (44.1%)

- New test cases that would fail with a runtime error for the old version of the program are added to validate bug fixes; new tests that would not compile against the old version of the program are likely added to validate new functionality.
- A significant number of new tests are added not necessarily to cover new code, but rather to exercise the changed parts of the code after the program is modified.

# Thanks

Questions?

